Содержание

Предисловие научного редактора

Уважаемый читатель!

На сегодняшний день перед лицом разработчиков высокопроизводительных вычислительных систем встает вопрос о выборе направления в развитии. Стандартные подходы по увеличению производительности почти исчерпали весь свой потенциал. Так, масштабирование размеров отдельных элементов близко к фундаментальным ограничениям, а увеличение частоты приводит к слишком высокому рассеиванию тепла и большому потреблению энергии. Все это заставляет искать новые решения по развитию вычислительных систем.

Данная книга позволяет читателю познакомиться с новой ветвью в развитии вычислительных систем – многопроцессорными системами на одном кристалле. Авторы книги знакомят читателей не только с работами передовых научно-исследовательских коллективов и академических сообществ, но также с результатами многолетнего труда ведущих компаний – изготовителей высокопроизводительных систем, таких как Motorolla, Intel, Nvidia и многих других.

В данной книге проводится обзор методов построения новых вычислительных систем, также представлены направления их развития, вводятся требования к системам межсоединений и системам арбитража, предлагаются принципы построения программного обеспечения. Что очень важно, на мой взгляд, – в книге представлены варианты построения систем с возможностью динамической подстройки под вычислительные нагрузки и предложены идеи по их программированию.

Несомненным достоинством данной книги является последовательность и удобство изложения материала, что делает ее удобной для чтения и обучения. Эта книга будет интересна для программистов, разработчиков встроенных систем, академических исследователей, энтузиастов электроники и всех тех, кто хочет познакомиться с миром многопроцессорных систем.

Немудров В.Г.
директор ФГУП «НИИМА «Прогресс»,
доктор технических наук, профессор

Предисловие

В следующее десятилетие, исходя из закона Мура, мы все еще рассчитываем на повышение плотности расположения транзисторов, объединяя миллиарды транзисторов на одном кристалле. Однако все более очевидным становится то, что применение большого объема параллелизма на уровне команд с глубокими конвейерами и более агрессивными суперскалярными технологиями широкого запуска операций исполняемой многооперационной команды, а также использованием основной части потенциала транзистора для кэш-памяти на кристалле зашло в тупик. В особенности становится все более затруднительной масштабируемая производительность с высокими тактовыми частотами из-за проблем с рассеянием тепла и слишком высокого потребления энергии. Последнее обстоятельство не только относится к техническим проблемам для мобильных систем, но даже может стать серьезной проблемой для вычислительных центров в связи с высоким уровнем потребления энергии, ведущим к значительным индексам стоимости в бюджете. Улучшение рабочих характеристик может быть достигнуто за счет использования параллелизма на всех уровнях системы.

Поэтому для высокопроизводительных вычислительных систем, для высокопроизводительных серверов, а также для встроенных систем происходит масштабное смещение концептуального подхода в отношении мультиядерных архитектур. Интеграция многочисленных ядер на одном кристалле ведет к существенному повышению производительности без увеличения тактовой частоты. Мультиядерные архитектуры обеспечивают большее отношение производительности к потребляемой мощности, чем одноядерные архитектуры с аналогичной производительностью.

Сочетание мультиядерной и сопроцессорной технологий обещает чрезвычайно высокую производительность выполнения расчетов для приложений с высоким потреблением времени ЦПУ, таких как научные расчеты и приложения специального назначения в области встроенных систем. Основанные на ППВМ программы ускоренной обработки не только дают возможность ускорения приложений за счет внедрения требующего большого объема вычислений ядер в аппаратные средства, но также используются для адаптации к динамическому поведению программ.

Целью этой книги является оценивание стратегий для разработки будущих систем в архитектуре многопроцессорных систем на одном кристалле (MPSoC). В книге будут рассматриваться оба аспекта: разработка аппаратных средств и интеграция инструментов в существующее развитие инструментов. Кроме того, к темам этой книги относятся новые тенденции в развитии MPSoC в сочетании с перенастраиваемой архитектурой. Основной акцент делается на архитектуры, ход проектирования, разработку инструментов, прикладных программ и проектирование систем.Нам приятно выразить благодарность авторам и соавторам за их выдающийся вклад, который они внесли в подготовку этой книги.

Кроме того, мы хотим выразить благодарность команде издательства Springer, а именно Аманде Дэвис, Чарльзу Глэйзеру и Еве Руби за их большую поддержку и терпение.

Юрген Бекер,
Майкл Хюбнер

г. Карлсруэ, Германия

ГЛАВА 1

ВВЕДЕНИЕ В МУЛЬТИЯДЕРНЫЕ СИСТЕМЫ НА ОДНОМ КРИСТАЛЛЕ: ТЕНДЕНЦИИ И ПРОБЛЕМЫ РАЗВИТИЯ

Лионель Торрес, Паскаль Бену, Жиль Сассателли, Майкл Роберт, Фабиан Клермиди и Диего Пуччини

1.1. От систем на одном кристалле (SoC) к многопроцессорным системам на одном кристалле (MPSoC)

Эмпирический закон Мура не только описывает увеличение плотности транзисторов, которое допускает технический прогресс. Он также выдвигает новые требования и порождает новые проблемы. Сложность систем увеличивается с такой же скоростью. Современные системы никогда не смогли бы быть спроектированы с применением тех же подходов, которые использовались 20 лет назад. Постоянно должны осваиваться новые архитектуры. Теперь стало ясно, что закон Мура для последних двадцати лет породил три основные революции. Первая революция, которая произошла в середине восьмидесятых, была способом встраивать все большее число электронных устройств в один и тот же кремниевый кристалл; это была эра систем на одном кристалле. Одна из основных проблем состояла в способе эффективного соединения всех этих элементов между собой (межсоединений). Для этих целей длительное время использовались шины межсоединений. И все же в середине девяностых промышленная и академическая общественность столкнулась с новой проблемой, когда число ядер для обработки данных стало слишком большим для их взаимного использования одной средой связи. Появились новые схемы межсоединений, которые основываются на сетевых матрицах связи, так называемые сети на кристалле; за последние десять лет интенсивные усилия по исследованиям привели к существенным улучшениям в этой области. Последний прорыв в начале 2000-х годов был обусловлен необходимостью межсоединений нескольких процессоров на одном и том же кристалле. Ранее разработанные системы использовали один встроенный процессор, управляющий кристаллом, а многочисленные управляющие процессоры теперь должны были распределять между собой централизованное управление. Появились первые многопроцессорные системы на одном кристалле (MPSoC) [1]. Они комбинировали несколько встроенных процес-

Л. Торрес (✉). Университет в Монпелье 2, UMR CNRS, Франция.
e-mail: Lionel.Torres@lirmm.fr.

соров, запоминающих устройств и специализированных схем (ускорителей, входов/выходов), которые соединялись между собой с помощью выделенных инфраструктур для обеспечения полностью интегрированной системы. В противоположность SoC MPSoC содержали два и более главных процессора, управляющих прикладным процессом, достигая повышенной производительности. С тех пор было разработано значительное число исследовательских и промышленных проектов [2]. Они стали появляться на рынке и, как ожидается, станут широкодоступными в еще большем разнообразии в следующие несколько лет [3].

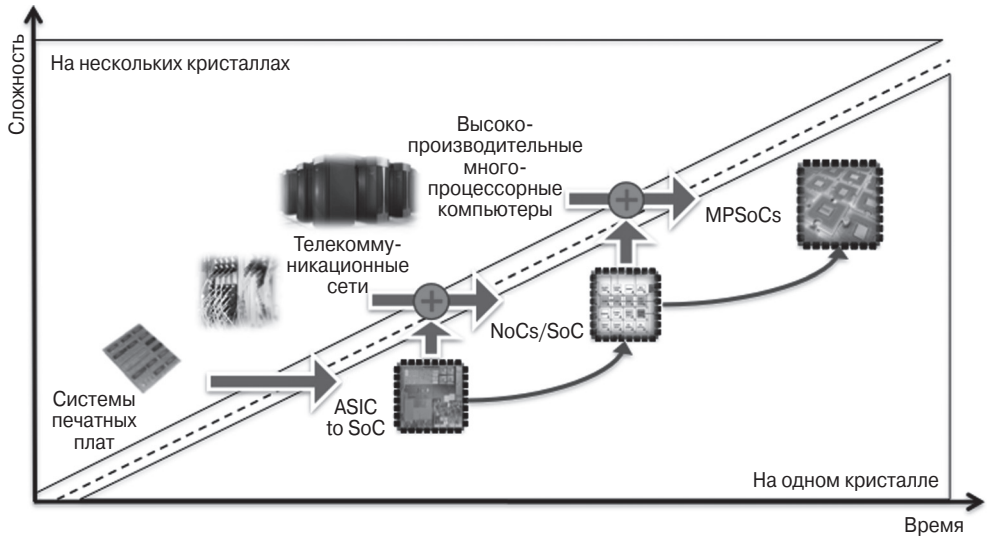


Рис. 1.1. От SoC к MPSoC

Теперь понятно, что эта третья революция радикально изменит способ применения архитектуры системы на кристалле. На рис. 1.1 подводится итог этих трех революций, которые произошли менее чем за 20 лет.

1.2. Общее устройство MPSoC

В этом разделе приводится описание типичной MPSoC, рассматриваются только ключевые элементы с целью формулировки обоснованных предположений относительно ее архитектуры. В общем случае MPSoC состоит из нескольких процессорных элементов (PE), связанных с помощью структуры межсоединений, которая приводится на рис. 1.2.

1.2.1. Процессорные элементы

Процессорные элементы MPSoC зависят от контекста прикладной программы и соответствующих требований. Мы различаем два семейства архитектур. С одной стороны, это гетерогенные MPSoCs, составленные из различных PE (процессоров, запоминающих устройств, ускорителей и периферийного оборудования).

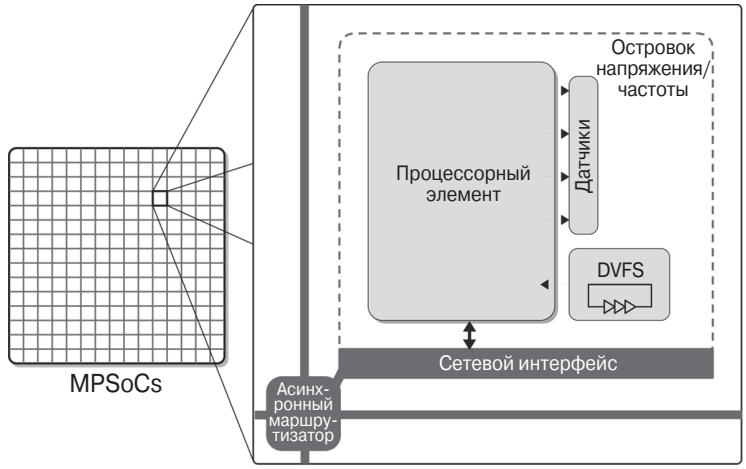


Рис. 1.2. Общая архитектура MPSoC

Эти платформы, безусловно, являются пионерами: сетевой процессор C-5 [4], Nexperia [5] и OMAP [6], представленные в [2]. Второе семейство представляет однородные многопроцессорные системы на одном кристалле (MPSoC), впервые использованные архитектурой Lucent Daytona [2, 7], в которой одна и та же плитка задействована несколько раз. Эта глава посвящена обоим семействам, и на рис. 1.2 представлена как однородная, так и неоднородная (гетерогенная) их структура. К примеру, в многочисленных работах считается, что процессоры, а также гибкие аппаратные средства, такие как перенастраиваемые матрицы, составляют гетерогенные PE.

1.2.2. Межсоединение

Описанные ранее процессорные элементы (PE) чаще всего соединяются между собой с помощью сети на кристалле (NoC) [8–11]. Сеть NoC включает в себя сетевые интерфейсы (NI), узлы маршрутизации и линии связи. NI осуществляет сопряжение между средой межсоединений и доменом PE. Он разделяет вычисления от функций связи. Узлы маршрутизации, которые также называют роутерами, которые несут ответственность за прокладку маршрута через эти линии связи, управляют передачей данных между исходящим и адресным PE. Были изучены несколько сетевых топологий [12, 13]. Рис. 1.2 представляет двумерную сетку межсоединений. Выбранный размер предложенной пропускной способности связи должен быть достаточным для набора адресного приложения.

NoC облегчают разработку глобально асинхронного локально синхронного (GALS) свойства путем использования в NI асинхронно-синхронных интерфейсов [14, 15]. На рис. 1.2 приведен пример асинхронного маршрутизатора для подчеркивания этого свойства.

1.2.3. Управление потреблением энергии

Одной из основных проблем для встроенных систем сегодня является обеспечение их энергоэффективности. Особенность GALS позволяет выполнять разбие-

ние MPSoC на несколько островков напряжения/частоты (VFI) [16]. В этом примере каждый VFI содержит PE, синхронизированный на заданную частоту и напряжение. Этот подход позволяет управлять потреблением энергии на мелкоструктурном уровне [17]. Как и в [18, 19], мы считаем, что MPSoC использует распределенное масштабирование динамического напряжения и частоты (DVFS): каждый PE включает устройство DVFS. Оптимизация использования энергии состоит в настройке напряжения и частоты каждого PE с целью балансировки расхода энергии и производительности. В более современных MPSoC используется набор датчиков, встроенных внутри каждого PE, которые обеспечивают информацией о потребляемой энергии, температуре, производительности или любом другом показателе, необходимом для управления DVFS. В любом случае из-за добавления выделенных схем во многих MPSoC используется управление электропитанием на более грубом структурном уровне, которое включает многочисленные PE в одном VFI, обеспечивая совсем другой уровень управления потреблением энергии.

1.3. Энергоэффективность и способность к перенастройке

Как уже упоминалось во введении, MPSoC подчиняются закону Мура [20]. Этот эмпирический закон демонстрирует свою правоту в течение нескольких последних десятилетий. На рис. 1.3 приведено несколько примеров процессоров с числом задействованных транзисторов. Что касается MPSoC, то с какими именно проблемами предстоит столкнуться по закону Мура? Большая плотность транзисторов означает также повышение производительности (но также и потребления энергии) из-за увеличения числа ядер. Это также означает повышение потребления энергии. В течение последних лет оптимизация потребления энергии стала одной из самых горячих тем проектирования не только устройств с питанием от батареи, но также большого разнообразия устройств в разных областях применения, от домашней электроники до высокопроизводительных компьютеров. ITRS [21] прогнозирует за следующие пять лет двукратное увеличение потребления энергии стационарных бытовых приборов (см. рис. 1.4). Более того, прогнозируется, что потери от утечек и динамическое потребление энергии для таких устройств будут эквивалентны для логической части и для запоминающих устройств. Эти тенденции в сочетании с требованием увеличения производительности превращаются для архитектуры MPSoC в настоящую проблему [5, 4]. Каким образом можно управлять балансировкой потребления энергии/производительностью в конструкциях с многомиллионными транзисторами?

Общепризнано, что современное управление потреблением энергии необходимо для достижения эффективности не только для мобильных устройств, но также и для всех видов электронного оборудования.

Если MPSoC проектируется как энергоэффективная, то ее рабочую среду нельзя рассматривать как статическую. Чтобы понять этот принцип, рассмотрим простой пример прикладных устройств связи четвертого поколения. Для поддержки высокой пропускной способности при плохом качестве каналов передачи дан-

ных требуются алгоритмы оценивания объединенных каналов, для которых нужен большой объем вычислений. Тем не менее, когда мобильный терминал подходит близко к базовой станции для экономии энергии, может быть использована простая схема. Как можно управлять этими изменениями в рабочей среде?

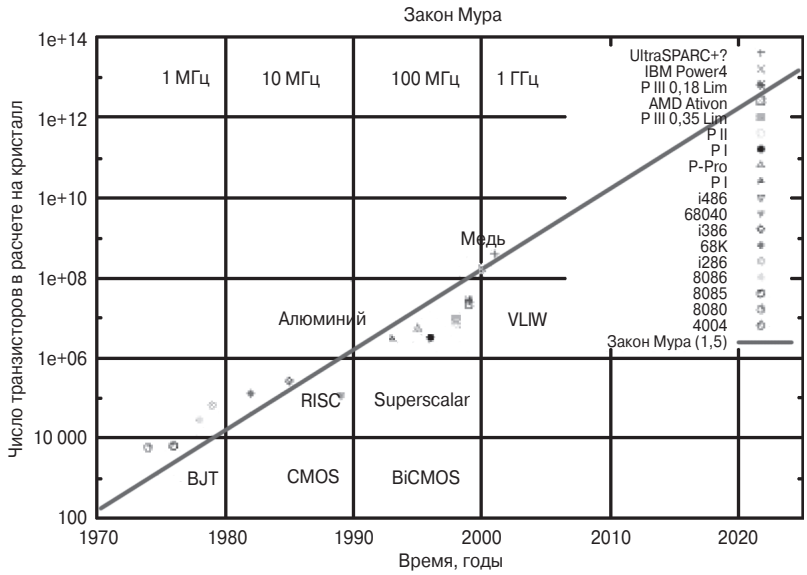


Рис. 1.3. Число транзисторов ЦПУ (http://www.ausairpower.net/moore-iw.pdf)

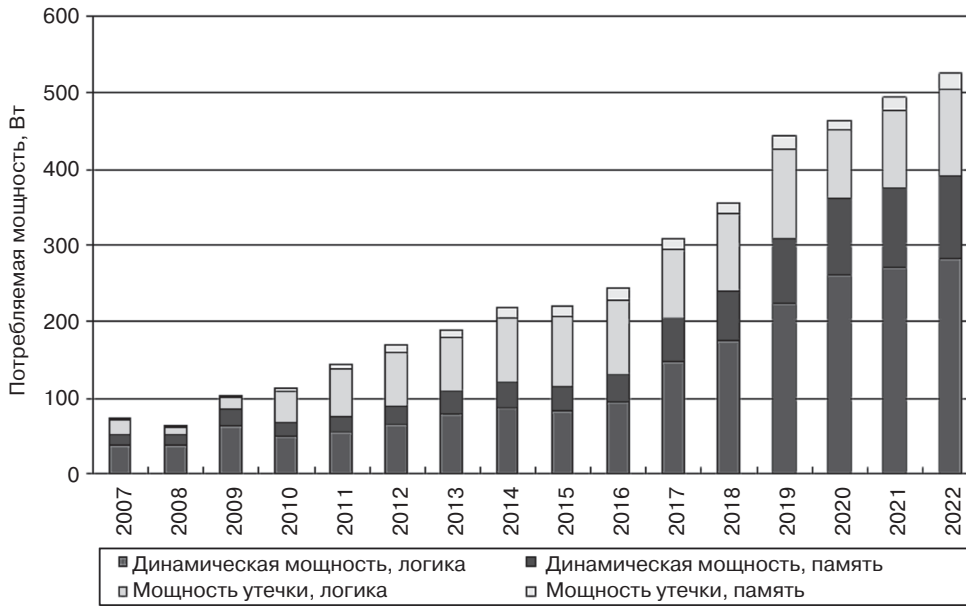


Рис. 1.4. Тенденции потребления энергии стационарным потребителем энергии SoC [21]

Второй пример условий окружающей среды учитывает технологическую неконтролируемость/невоспроизводимость. Закон Мура прогнозирует, что всевозрастающее число транзисторов с улучшенными рабочими характеристиками также несет с собой проблемы неконтролируемости. Неконтролируемость технологического процесса – это явление, всегда присутствующее в процессе изготовления КМОП-транзисторов, которое исторически всегда учитывалось с помощью проектного запаса с учетом статистики расхождения между кристаллами кристаллической пластины для интегральной схемы. И по мере сокращения размера транзисторов и разрастания этого явления копирование с изменчивостью стало настоящей проблемой: описание параметров в пределах одного и того же кристалла теперь стало неоспоримо влиять на работу системы. Под это влияние попали и MPSoC. Например, не все процессорные элементы одной и той же системы в состоянии работать на одной и той же тактовой частоте. В результате этого два экземпляра одной и той же MPSoC часто обладают разными уровнями производительности. Поэтому как могут разработчики гарантировать управление рабочими параметрами при такой их изменчивости в процессе изготовления?

Для улучшения энергоэффективности в динамических средах при наличии неконтролируемости может быть использована способность к самостоятельной адаптации. Другими словами, выходом из этого тупика может служить система, способная к самостоятельной настройке в соответствии с изменениями среды в целом или элементов самой системы с целью выполнения соответствующих требований.

1.4. Усложнение и масштабируемость

Как утверждалось во введении, преимущества, предсказанные законом Мура, способствуют усложнению систем за счет роста числа используемых процессорных элементов. Для того чтобы проиллюстрировать это усложнение, на рис. 1.5 приведены тенденции, которые были получены ITRS [21]: число процессорных ядер в SoC портативной бытовой аппаратуры увеличится примерно в 3,5 раза за следующие пять лет. Более того, размеры памяти и логических устройств будут следовать той же тенденции. В этом контексте можно ли будет управлять более чем шестью сотнями процессоров, появление которых прогнозируется через следующие 10 лет?

Существует проблема, которая лежит в основе преграды в виде сложности системы – масштабируемость. Масштабируемость является свойством системы, которое указывает на ее способность менять масштаб для воплощений в больших размерах. В случае MPSoC это относится к возможности системы увеличивать суммарную вычислительную способность при добавлении ресурсов. Систему, чьи рабочие характеристики улучшаются после добавления аппаратных средств пропорционально производительности добавленных средств, называют масштабируемой системой. Говорят, что алгоритм, конструкция, сетевой протокол, программа или иная система масштабируются, если они не теряют своей эффективности и целесообразны для применения к более крупным объектам.

Рис. 1.5. Тенденции усложнения SoC портативных конструкций [21]

То, что можно считать хорошим решением сегодня, возможно, не будет хорошим решением уже завтра: конструкции, основанные на платформе и повторном использовании ядра, подвели проектировщиков промышленных систем к некоторым вполне понятным соображениям в отношении производительности и рабочих характеристик. Эти методы проектирования вызывают все больше сомнений и в дальнейшем могут больше не масштабироваться. Одним из основных препятствий является то, что эти решения плохо масштабируются в терминах программного обеспечения и аппаратных средств. Мы твердо верим, что альтернатива возможна на основе масштабируемых аппаратных средств и программного обеспечения. Поэтому распределение функций управления MPSoC является очень важной проблемой.

1.5. Неоднородный и однородный подходы

В контексте использования масштабируемости, связанной с необходимостью самостоятельной адаптации, системы MPSoC становятся все более популярным решением, которое сочетает гибкость программного обеспечения наряду с возможностью существенного повышения быстродействия. Как уже отмечалось во введении, мы будем различать между собой:

– неоднородные MPSoC, также называемые многопроцессорным чипом или многоядерными системами: эти системы состоят из процессорных элементов различных типов, таких как один или несколько универсального типа процессоров, процессор для цифровой обработки сигналов (DSP), аппаратные ускорители, периферийное оборудование и модуль межсоединений наподобие NoC;

– однородные MPSoC, которые в этом подходе являются базовыми процессорными элементами со всеми встроенными элементами, требуемыми для SoC: один или несколько процессоров (универсального типа или специализированные), запоминающее устройство и периферийное оборудование. Эта плитка затем несколько раз задействуется, и все эти элементы соединяются между собой посредством специализированной инфраструктуры связи.

По существу, первый подход предлагает наилучший компромисс между производительностью и потреблением энергии, а второй, очевидно, является более гибким и масштабируемым, но в меньшей степени энергоэффективен. Благодаря своей высокой энергоэффективности разнородные MPSoC используются для портативных систем и более универсальных встроенных систем, тогда как однородный подход чаще всего используется в видеоигровых приставках, настольных компьютерах, серверах и при обработке данных на сверхмощных ЭВМ.

1.5.1. Неоднородные MPSoC

Неоднородные MPSoC представляют собой комплект соединенных между собой ядер с различными функциональными возможностями. На рис. 1.6 приводится обзор характерных неоднородных MPSoC, состоящих из набора универсальных процессоров (ЦПУ), нескольких ускорителей (видео, аудио и т.д.), элементов памяти, периферийных устройств и инфраструктуры межсоединений.

За рамками аппаратной архитектуры система MPSoC обычно управляет работой нескольких прикладных программ, разделенных на задачи, и операционной системой, управляющей в свою очередь аппаратными и программными средствами с помощью программного обеспечения промежуточного слоя (т.е. драйверами). Рис. 1.7 иллюстрирует схематический вид MPSoC и взаимодействие между программными и аппаратными средствами.

Для того чтобы проиллюстрировать общие принципы, изложенные в предыдущем разделе, мы можем сослаться на Nexperia Philips, или на ST Nomadik, или на хорошо известную платформу TI OMAP, или MPSoC MORPHEUS [22] из Европейского проекта MORPHEUS. Функциональная и структурная неоднородность этих платформ позволяет получать высокую производительность и энергоэффективность, давая возможность их интеграции в портативные устройства, такие как мобильные телефоны.

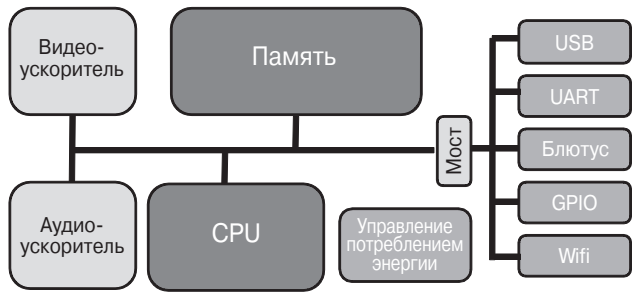


Рис. 1.6. Упрощенный обзор разнородных MPSoC

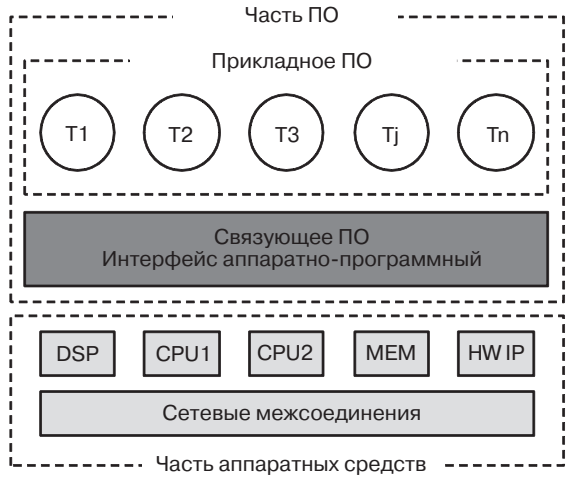


Рис. 1.7. Схематичный вид MPSoC

Термин «платформа» также придает некоторую гибкость этому подходу. Действительно, с помощью платформ можно настроить систему под специальные прикладные программы благодаря базовой инфраструктуре «процессор – запоминающее устройство – шина» и библиотеке дополнительных ускорителей и периферийного оборудования. Этот подход дает возможность сократить единовременные затраты на проектирование и время, необходимое продукции для выхода на рынок, но также несет с собой ряд недостатков. Его гибкость ограничивается фазой проектирования или в некоторой степени периодом после изготовления, поскольку специальные функциональные способности ускорителей невозможно перенастраивать. Масштабируемость также является проблемой для таких платформ, поскольку требуемая полоса пропускания канала связи зависит от количества и типов ускорителей, и, таким образом, может требоваться некоторая адаптация для каждой конкретной конструкции.

1.5.2. Однородная MPSoC

Как уже отмечалось в предыдущем разделе, разнородные системы MPSoC обеспечивают сегодня наилучший компромисс между производительностью и энергоэффективностью и являются оптимальным решением для встроенных систем, но и они также страдают от ограниченной степени гибкости и масштабируемости.

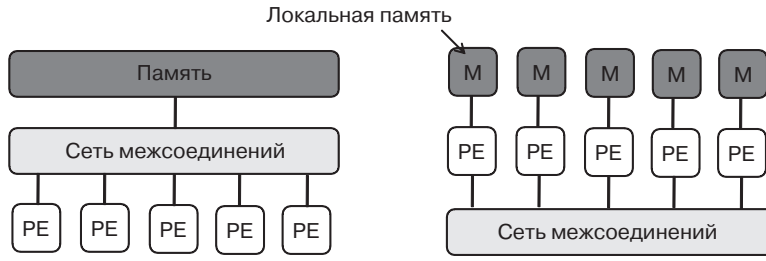
Альтернатива лежит в построении однородной системы на базе одного и того же программируемого блока, несколько раз подвергаемого обработке. Эту модель архитектуры в литературе часто называют моделью параллельной архитектуры. Параллельные архитектуры особенно обстоятельно изучались в течение последних 40 лет теорией вычислительных систем и проектированием ЭВМ. В настоящее время интерес к подходам подобного рода растет в связи с использованием встраиваемых систем. Основной принцип архитектуры, которая демонстрирует возможности параллельной обработки данных, основывается на

возрастании числа физических ресурсов для распределения между ними необходимого времени выполнения. Теоретически архитектура, выполненная на N процессорных элементах, может обеспечить повышение быстродействия почти в N раз; однако это увеличение быстродействия сложно (или невозможно) получить на практике. Другое преимущество использования многопроцессорных элементов перед однопроцессорными системами состоит в том, что можно соответствующим образом понижать частоту и, как следствие, величину напряжения питания: поскольку потребляемая энергия связана с напряжением квадратичной зависимостью, то это снижение динамического потребления электроэнергии оказывается значительным. Динамическое потребление мощности: $P_{дин} = \alpha \cdot C_{нагр} \cdot V_{DD}{}^2 \cdot F_{такт}$, где $P_{дин}$ является динамическим потреблением мощности, α – коэффициент использования цепи, т.е. коэффициент коммутирующей цепи, $C_{нагр}$ – эквивалентная емкостью цепи, V_{DD} – напряжение питания, а $F_{такт}$ – тактовая частота. Предполагая возможность уменьшения тактовой частоты на долю r ($0 < r < 1$), можно также рассчитывать на такую же долю уменьшения напряжения питания благодаря использованию метода динамического масштабирования напряжения и частоты (DVFS). И окончательно динамическое потребление энергии: $P_{дин} = \alpha \cdot C_{нагр} \cdot (r \cdot V_{DD})^2 \cdot (r \cdot F_{такт})$; при $r = 0{,}8$ динамическое потреблений электроэнергии практически уменьшается вдвое.

Однородные MPSoC, основанные на программируемых параллельных процессах, могут обеспечить производительность благодаря «увеличению быстродействия» и сокращению потребления электроэнергии при снижении рабочей частоты и подачи электроэнергии и могут рассматриваться как реальная альтернатива разнородным MPSoC. Кроме того, присущая им структура обладает большей гибкостью, большей степенью масштабируемости по сравнению с разнородными системами. Эффективное использование параллельности на практике, однако, не является простой задачей; гибкость и масштабируемость также могут быть ограничены из-за нескольких факторов, таких как организация памяти, инфраструктура межсоединений и т.п.

Параллельные архитектуры интенсивно изучались в течение последних 40 лет; в результате имеется большое количество книг и библиографий, связанных с этим вопросом, и поэтому мы сделаем акцент только на общих принципах.

Первая знаменитая классификация была предложена Флинном [23]. Он классифицирует архитектуры в соответствии со связью между обрабатывающими и управляющими блоками. Он определяет четыре модели выполнения: SISD (один поток команд и один поток данных), SIMD (один поток команд, множество потоков данных), MISD (с одним потоком данных и множеством потоков команд) и MIMD (множество потоков данных и множество потоков данных). Модель SISD является классической моделью фон Неймана [24], в которой один процессорный ресурс, выполняющий одну команду в единицу времени, обрабатывает один поток данных. В архитектуре SIMD один управляющий блок использует все потоки данных и распределяет в них данные для каждого обрабатывающего ресурса. Архитектура MISD одновременно выполняет несколько команд на одном потоке данных. И, наконец, несколько блоков управления управляют несколькими блоками обработки данных в архитектурах MIMD.



а) Совместно используемая память *б)* Распределенная память

Рис. 1.8. Организация памяти

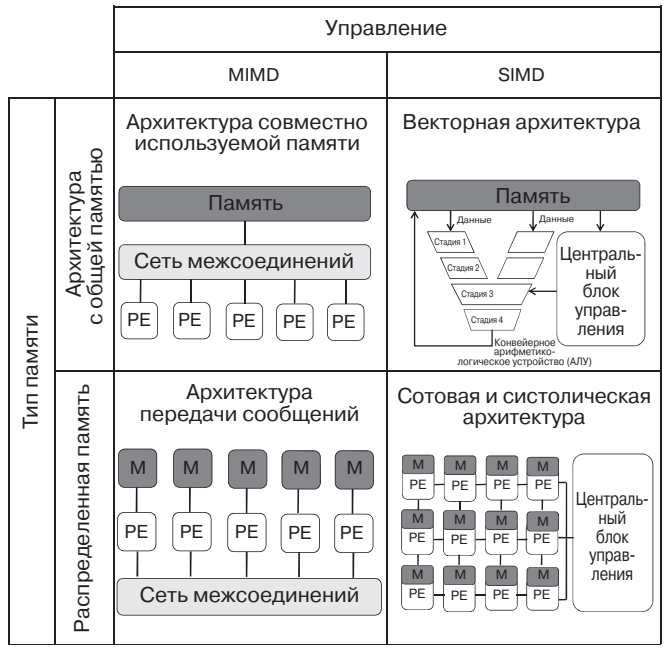


Рис. 1.9. Систематическая классификация архитектуры

На рис. 1.8 представлена расширенная классификация Флинна с учетом организации памяти, которая может быть использоваться совместно (рис. 1.8*а*) или быть распределена (рис. 1.8*б*). В архитектуре совместно используемой памяти процессы (выполняемые различными процессорами) могут легко обмениваться информацией через совместно используемые переменные; однако для этой архитектуры требуются осторожное обращение с синхронизацией и защита памяти.

В архитектуре распределенной памяти требуется инфраструктура связи для соединения обрабатывающих элементов и их запоминающих устройств, которое даст возможность обмениваться информацией.

На основе организации памяти и управления рис. 1.9 отображает классификацию архитектур с параллельной обработкой данных. В этой классификации раз-

деляются централизованное (SIMD) и децентрализованное управление (MIMD), а также совместно используемая и распределенная память.

Важно обратить внимание, что организация управления и памяти обеспечивает различные компромиссы между масштабируемостью и управляемостью системы. Например, архитектура на основе полностью распределенного управления и организации памяти будет обеспечивать большую степень масштабируемости, но меньшую степень гибкости в управлении, чем архитектура, основанная на централизованном управлении и совместном использовании памятью.

1.6. Оптимизация при наличии многих переменных

Благодаря повышению сложности архитектуры MPSoC ее оптимизация стала настоящей проблемой, поскольку она может преследовать многочисленные противоположные цели: производительность прикладной программы, потребление электроэнергии, температуру, балансирование нагрузки и т.п. В литературе описано несколько разработанных методов для решения этой проблемы. Классические подходы являются статическими и не нацелены на оптимизацию системы во время ее разработки. Последние методы применяются в оперативном режиме и пытаются адаптировать систему динамически. Наиболее современные методы стремятся использовать преимущества распределенного решения процессорных элементов для повышения масштабируемости системы.

1.6.1. Статическая оптимизация

Что касается MPSoC, то статическая оптимизация является способом улучшения системы во время проектирования. Несколько авторов предложили метод статической оптимизации для повышения энергоэффективности. Например, в [25] авторы используют генетические алгоритмы для решения задачи по оптимизации системы в процессе ее проектирования. Они исследовали метрики, включающие трафик линии связи, коэффициент использования памяти и пропускную способность.

В [26] авторы анализируют три статических метода оптимизации: «жадный» алгоритм, алгоритм запрещенного поиска и метод имитации отжига. Построение графика выполнения задач при минимальном потреблении электроэнергии и соблюдении некоторых временных ограничений изучается как часть процесса использования пространства проектных параметров. Во-первых, описание системы разлагается на графы синхронного потока данных (SDF) [27] в одночастотной области, включающей временные ограничения. Затем предлагается расширение традиционного SDF до графов многочастотной области.

В [28] статический подход, основанный на линейных моделях, предлагается для оптимизации потребления электроэнергии при соблюдении ограничений в режиме реального времени. Рассмотрена задача выбора наилучшей рабочей частоты для каждого блока распределенного проектирования. Автор моделирует набор фреймовых конвейерных приложений, используя графы SDF. Затем приложения наносятся на распределенную платформу, интегрирующую мелкоструктурные DVFS.

1.6.2. Динамическая оптимизация

При статической оптимизации всегда требуется принимать решения на этапе проектирования. Однако с учетом роста неопределенности технологий реализации и используемых сценариев таких систем динамическая оптимизация становится необходимой для осуществления гибких подходов и надежных конструкций [29]. Централизованные и распределенные подходы излагаются в следующих разделах.

1.6.2.1. Централизованные подходы

В противоположность статической оптимизации, динамические подходы предоставляют возможность приспособления. На рис. 1.10 приведен схематический вид распространенных динамических подходов, используемых для MPSoC, – оптимизация централизованной подсистемы, которая отвечает за управление системой в целом. Она анализирует глобальную информацию и оптимизирует каждый процессорный элемент в системе.

В работе [30] рассматривается выбор частоты и напряжения для систем GALS на основе VFI. Централизованный метод, основанный на нелинейной оптимизации Лагранжа, используется для выбора частот и напряжений. Они представляют статический и динамический алгоритмы. Кроме того, авторы утверждают, что в системах со скрытыми ограничениями определение оптимальных значений напряжения потребует решений глобальной стратегии.

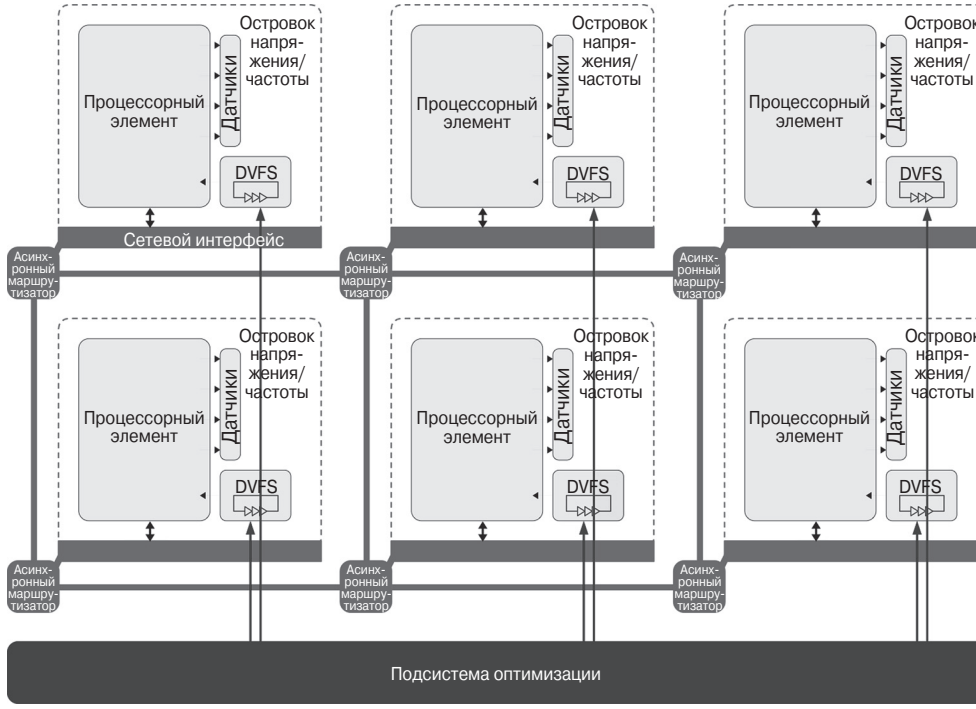


Рис. 1.10. Централизованная динамическая оптимизация на MPSoC

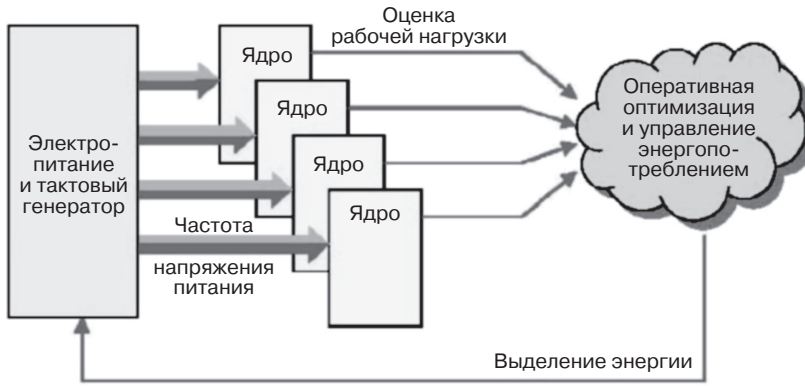


Рис. 1.11. Централизованное оперативное глобальное управление энергией [32]

Аналогичным образом в [31, 32] авторы предлагают централизованное управление потреблением электроэнергии с помощью моделей, основанных на законе Кирхгоффа для тока. Они показали, что если локальное рассеяние тепла каждого процессорного элемента можно минимизировать с помощью методов DVS, основанных на прогнозируемых значениях рабочей нагрузки, то эти локальные минимумы обычно не совпадают с положением глобального минимума.

Кроме того, глобальный минимум может быть получен с учетом зависимостей относительного отсчета времени всех задач, выполняемых в системе. Их подход основывается на блоке оперативного глобального управления потребляемой энергией, который контролирует процессорные элементы с помощью источника питания и тактового генератора. Блок-схема этого подхода приводится на рис. 1.11. Авторы используют аналогию между задачей минимизации потребления энергии при наличии временных ограничений в общем графе и задачей минимизации потребления энергии при наличии ограничений в виде закона Кирхгоффа для тока в эквивалентной резистивной цепи.

В работе [33] авторы используют выпуклую оптимизацию для присваивания частоты с учетом температуры на MPSoC. Во-первых, они представляют температурную модель всей системы. Затем эта задача формулируется для установившегося и динамического состояний: назначение частоты каждому процессору, поддержание температуры и потребления энергии ниже установленных пользователем пороговых значений. В установившемся состоянии частота и напряжение назначаются один раз и остаются постоянными, не используя преимущества DVFS. В динамическом состоянии частота и напряжение изменяются со временем для лучшей оптимизации производительности системы.

Авторы формулируют оба сценария как задачи выпуклой оптимизации. Затем они предлагают процедуры оптимизации для установившегося и динамического состояний. Для случая динамического состояния используется двухэтапный алгоритм. При этом авторы только приводят математическую формулировку процедуры. Используя программу Matlab для выпуклой оптимизации, они показали в этой работе решение иллюстративного сценария. Те же авторы предлагают в [34] с помощью метода выпуклой оптимизации предварительно рассчиты-

вать некоторые значимые решения на этапе проектирования и осуществлять контроль для оперативного выбора наилучшего решения для каждого случая в процессе работы. На рис. 1.12 приводится схема для составления таблицы в период проектирования.

Рис. 1.12. Составление таблиц в период проектирования с помощью выпуклой оптимизации [34]

В [35] авторы анализируют некоторые исследования для энергоэффективного планирования в системах реального времени на платформах, интегрирующих DVFS. После длительного анализа используемых методов для однопроцессорных систем в этой статье многопроцессорные платформы делятся на однородные и неоднородные. Для платформ первого типа кратко описываются некоторые методы, применяемые для планирования фреймовых задач в режиме реального времени, периодического планирования в режиме реального времени, планирования энергоэффективности с учетом утечек и планирования использования резервного времени.

Для случая неоднородных систем представлено несколько методов для периодических задач реального времени и минимизации затрат при ограничениях на энергопотребление.

В [36] обсуждается эвристический подход для оптимального расположения задач в неоднородных MPSoC на базе NoC. Существует несколько подходов, в которых задачи перемещаются для уравновешивания рабочей нагрузки вычислений и равномерного рассеяния энергии, как это описано, например, в [37]. В некотором смысле, согласно [38], исключение горячих точек и регулирование теплового градиента стало важной задачей оптимизации в отношении надежности MPSoC.

В [38] авторами анализируются динамические программы планирования задач и распределения ресурсов на основе операционных систем (ОС), которые используются для управления тепловым режимом MPSoC. В [39], кроме того, минимизируются тепловые градиенты. Они делают акцент на MPSoC, в которых рабочая нагрузка неизвестна априори и в общем случае нелегко предсказуема. При этом предлагается основанная на ОС политика перемещения задач и планирования работы, которая оптимизирует тепловой профиль чипа за счет балансировки системной нагрузки. Авторы заявляют о значительном сокращении временных и пространственных колебаний температуры.

В [40, 41] авторы предлагают в период проектирования определять оптимальные по Парето характеристики в сочетании с оперативным управлением. Во-первых, они предлагают выполнить многомерную оптимизацию на этапе проектирования. Многомерное пространство включает затраты (например, на потребление энергии), ограничения (например, на производительность) и задействованные ресурсы платформы (например, используемую память, процессоры, тактовые генераторы, полосу пропускания канала связи). Несложные устройства управления процессом работы, используемые ОС, принимают наиболее важные решения во время второй фазы.

В [42] рассматривается, как можно оперативно выбрать энергоэффективное распределение ресурсов на неоднородных многопроцессорных платформах. С учетом того, что могут быть различные варианты реализации одной и той же прикладной программы и что выбранный вариант должен удовлетворять предельным срокам ее выполнения при имеющихся ресурсах платформы, авторы предлагают использовать NP-трудную модель для решения проблемы выбора – так называемую многомерную многовариантную задачу о ранце. Для нахождения решения, близкого к оптимальному, они предложили использование ОС на эвристической основе.

1.6.2.2. Распределенные подходы

С прогнозируемыми сотнями процессорных элементов (PE) масштабируемость также является значительной проблемой для оптимизации процесса. По этой причине предлагается альтернативный подход для оптимизации процесса с помощью распределения ресурсов. Статическая оптимизация не обеспечивает адаптивность в процессе работы. В противоположность ей существующие динамические подходы обеспечивают способность реагирования во время работы на смену условий, но они являются централизованными решениями. Они не в состоянии обеспечить масштабирование, поскольку они не основываются на распределенных решениях.

Альтернативными для централизованных подходов являются решения, которые учитывают распределенные алгоритмы. Один интересный подход состоит в использовании архитектуры, представленной на рис. 1.13: каждый процессорный элемент MPSoC использует встроенную подсистему оптимизации, которая основывается на распределенном алгоритме. Эта подсистема управляет локальными исполнительными механизмами (на этом рис. это DVFS), учитывая рабочие условия. Другими словами, цель состоит в использовании распределенного алгоритма динамической оптимизации.

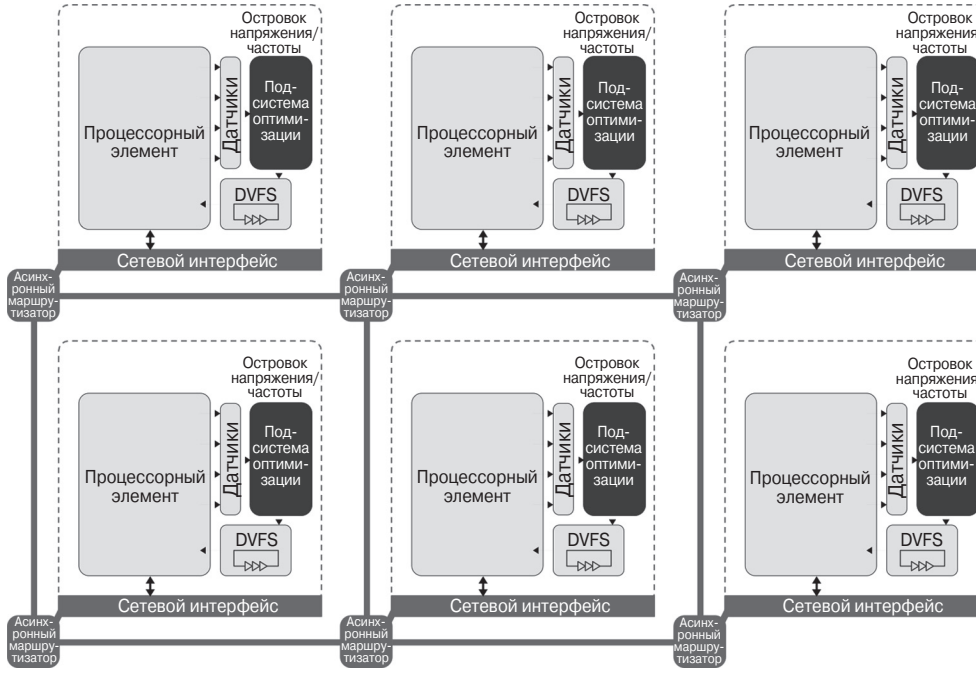


Рис. 1.13. Распределенная динамическая оптимизация на MPSoC

Во избежание образования точек с повышенной температурой и регулирования температуры плиток динамическое масштабирование пары напряжение–частота (DVFS) может быть использовано на уровне процессорных элементов. На системном уровне это подразумевает динамическое управление различными парами напряжений–частот каждого процессорного элемента с целью получения глобальной оптимизации. В работе [43] представлен оригинальный подход на базе теории игр, в котором происходит оперативная настройка частоты каждого процессорного элемента. Он преследует цель уменьшить температуру плитки при одновременном сохранении синхронизации между задачами графа приложения. Предполагается использование полностью распределенной схемы для возможности построения масштабируемого механизма. Результаты показывают, что предложенный оперативный алгоритм находит решения в нескольких вычислительных циклах, достигая снижения температуры на 23%. В [44] приведенные результаты показывают, что предложенный алгоритм оперативной оптимизации требует в среднем 20 вычислительных циклов, для того чтобы найти решение для 100-процессорной платформы, и достигает при этом эквивалентных рабочих показателей при сравнении с автономным методом.

В работе [45] этот адаптивный метод используется для сокращения потребления электроэнергии. Он оптимизирует частоты локальных процессоров при выполнении использующихся оперативных ограничений. Полученный выигрыш в потреблении электроэнергии при проведении теста связи составил от 10 до 25%, тогда как время отклика на временные изменения, вызванные перестройкой конфигурации прикладной программы, составило менее 25 мс.

1.7. Статическая и динамическая оптимизации в централизованном и распределенном подходах

Табл. 1.1 подытоживает методы оптимизации, описанные в предыдущем разделе. Было рассмотрено несколько подходов, представляющих направления оптимизации MPSoC. Таблица позволяет производить количественное сравнение разных методов. Эти методы сравниваются в отношении автономной и динамической фаз, их сложности и их осуществления (централизованный или распределенный подходы). Подходы, представленные в [34, 33] и [30], имеют сложную фазу автономной оптимизации. В [40, 41, 42] основная часть оптимизации проводится на этапе проектирования. Эти подходы едва ли можно использовать оперативно из-за их очень высокой сложности.

Решение, предложенное в [30, 32], использует подсистемы динамической оптимизации с низким уровнем сложности. Аналогичным образом подходы [38, 39] и [40, 41, 42] обеспечивают управление во время работы программ. Тем не менее, все эти подходы не справляются с учетом распределения. Когда используются ва-

Таблица 1.1. Обобщение оптимизаций MPSoC

	EPFL	**Stanford & EPFL**	**Carnegie Melon**	**SUN&U, California**	**IMEC**	**LKMM & CEA LETI**
Ссылка	[32, 33]	[34, 35]	[31]	[39, 40]	[41, 42, 43]	[43, 44, 45]
Модель	Аналог закона Кирхгоффа для силы тока	Выпуклая оптимизация	Нелинейная оптимизация Лагранжа	Целочисленное линейное программирование	Автономное обследование	Теория игр
Автономная фаза	Да	Да	Да	Нет	Да	Нет
Сложность	Средняя	Высокая	Высокая	–	Очень высокая	–
Динамическая фаза	Да	–	–	Да	Да	Да
Сложность	Низкая	–	–	Средняя	Средняя	Низкая
Распределенное использование	Нет	Нет	Нет	Вероятное	Вероятное	Да
Распределенная модель	Нет	Нет	Нет	Нет	Нет	Да
Целевые показатели	Энергопотребление	Производительность	Энергопотребление, пропускная способность	Горячие точки, градиент температуры	Многочисленные	Многочисленные
Ограничения	Скрытые	Температура	Скрытые	–	–	Скрытые, энергопотребление
Исполнительный механизм	Глобальное управление энергопотреблением	DVFS	Напряжение и частота	Планирование перемещения задач на базе ОС	На базе ОС	Локальное DVFS

рианты реализации на основе ОС ([38, 39] и [41, 42, 43]), становится возможным использовать распределенную систему. Однако мы не будем рассматривать такие подходы, поскольку они не основываются на распределенных моделях. Подход, основанный на теории игр [44, 45], по своей природе основывается на распределенной модели, которая улучшает масштабируемость системы. Более того, этот низкого уровня сложности метод может быть легко использован в оперативном режиме, который подразумевает хорошую адаптивность, требуемую динамическими системами.

И, наконец, табл. 1.1 также сравнивает количественные показатели, использованные в каждом случае. Несложно заметить, что далеко не каждый подход включает накладываемые ограничения, но все они предлагают многоцелевую оптимизацию. Некоторые из этих моделей могут быть использованы повторно в новых формулировках задач оптимизации.

1.8. Выводы

В настоящее время полупроводники претерпевают глубокие изменения, вызванные несколькими факторами, такими как приближение к границам возможностей кремниевых КМОП-технологий, а также неадекватность компьютерных моделей, использованных до настоящего времени. Эти изменения требуют разработки новых подходов в проектировании и программировании будущих интегральных схем. Поэтому параллелизм представляется единственным решением в борьбе с всевозрастающим требованием повышения производительности. Решения, предлагаемые в литературе, часто предполагают наличие у системы возможностей принимать оперативные решения для преодоления этих проблем, таких как масштабирование напряжения питания и частоты для повышения энергоэффективности, или тестирования цепи для выявления неисправных компонент и исключения их из функциональных ресурсов.

Конечно же, системы MPSoC являются естественной целью для внедрения этих технологий в практику: при условии соблюдения ими некоторых проектных норм они в состоянии обеспечить масштабирование с точки зрения рабочих характеристик. Более того, поскольку они по существу представляют собой распределенные архитектуры, то они также будут приспосабливаться к локально контролируемым и регулируемым параметрам системы.

В этой главе были рассмотрены многопроцессорные системы и сделан обзор модели, которая, как мы надеемся, представляет собой завтрашние MPSoC-системы. Из важных характеристик, которые были рассмотрены нами, наиболее важными являются гибкость, масштабируемость и способность к адаптации, если рассматривать децентрализованное управление, однородные или неоднородные матрицы процессорных элементов, распределенную память, масштабируемую сеть связи NoC-типа.

И, наконец, мы считаем, что способность к адаптации в ближайшем будущем найдет самое широкое применение в этой области. И не только из-за упоминавшихся здесь ограничений, таких как метод линейного сжатия, потребление энергии и надежность, но также потому, что использование компьютеров, несомненно, становится всеобъемлющим. Всеобъемлющее использование компьютеров или